Small Field-of-View Star Identification Using Bayesian Decision Theory

Daniel S. Clouse, Curtis W. Padgett

Jet Propulsion Laboratory, California Institute of Technology
Pasadena, California 91109-8099
and
The University of California, San Diego
Computer Science and Engineering Dept.

Abstract

In this document, we describe a simple autonomous star identification algorithm which is effective using a narrow field of view (2 degrees), making the use of a science camera for star identification feasible. This work extends that of Padgett and Kreutz-Delgado [8] by setting decision thresholds using Bayesian decision theory. Our simulations show that when positional accuracy of imaged stars is 0.5 pixel (standard deviation) and the apparent brightness deviates by 0.8 unit stellar magnitude, the algorithm correctly identifies 96.0% of the sensor orientations, with less than a 0.3% rate of false positives.

1 Introduction

A reliable attitude determination system is critical to the success of a space mission. A spacecraft that does not know its orientation in space is unable to send collected data back to Earth, receive instructions, or perform maneuvers. An attitude determination system should have the ability to estimate the orientation of the spacecraft with high precision and autonomously recover the attitude knowledge if it is lost or found to be unreliable. Considering the amount of time and money invested in a space mission, this is a very important problem.

A number of instruments such as GPS, and magnetometers are well suited for supplying attitude information while in orbit around Earth. A horizon sensor may also provide useful information while in the vicinity of any large body. However, these instruments are not very useful while the spacecraft is in transit between bodies. There are several other instruments that can supply attitude information to spacecraft control systems during these extensive transit times. Gyroscopes, sun sensors, and star sensors are generally integrated with on-board systems to provide feedback regarding spacecraft angular displacement. The signal from these instruments can be used to update and monitor changes in the orientation of the spacecraft with respect to some inertial reference frame [14].

Each of the transit sensors has unique advantages and disadvantages. Gyros provide continuous and nearly instantaneous information, but that information is relative, and therefore, gyros are of no use when the current attitude knowledge is lost. Sun sensors are inexpensive, lightweight, but require slewing of the sensor, or possibly the entire spacecraft, in order to find the target, the sun. Also, a sun sensor provides only 2-axis information, and therefore must be used in conjunction with some other instrument to completely determine spacecraft orientation. In addition, for deep space missions, finding the sun may not be practical.

On the other hand, star sensors are the most accurate of the sensors and allow for attitude estimation without prior information, or slewing the sensor. Unlike sun sensors, star sensors can also provide 3-axis information. Also, if slewing is possible, several attitude estimates taken from different parts of the sky may be combined to improve the overall accuracy. For the great majority of interplanetary missions, star fields are visible in nearly all orientations. Therefore, a fixed sensor can image an arbitrary section

of sky and determine the correspondence between the imaged stars and a catalog of reference stars stored on board. A star sensor can also be used to track known stars, providing continuous and extremely accurate estimates of angular displacement. Unfortunately, star sensors are typically heavy, expensive, and consume more power than other alternatives.

Since most missions already carry a science camera for acquiring highly accurate visual images of target bodies, an attractive alternative to a specialized star sensor is to use the science camera to acquire a star field image and use it for attitude estimation and star tracking. This would avoid most of the drawbacks of the star sensor while maintaining the accuracy advantage. Fewer instruments decrease the weight and complexity of the spacecraft, thus reducing the cost of the mission. This is an important advantage for proposed microspacecraft missions.

It is, in fact, quite possible that using the science camera rather than a star sensor will improve the accuracy of the attitude estimation. A science camera generally has a very small field of view (FOV) (2 degrees or less) compared to a specialized star tracker (15 degrees or more). A narrower FOV means each pixel covers less area. Consequently, we expect the final attitude estimate returned from a science camera image to be more accurate in pitch and yaw than would be possible with a standard star tracker. Also, because the same instrument is used for imaging and for finding the orientation, the error introduced by inaccurate alignment between two separate instruments is removed.

Unfortunately, the small FOV of the science camera also introduces problems. The density of bright stars across the sky is generally insufficient to insure a minimum number of stars in all potential sensor orientations for such a small area. For this scheme to work, a longer exposure will be required to increase the total number of visible stars. The on-board star catalog will need to be expanded to accommodate these new, dimmer stars, and this will require more on-board memory. In addition to increasing the amount of on-board memory, the incorporation of more stars also increases the difficulty of identifying any particular star, since now there are many more possibilities which must be distinguished. Thus, the star identification problem becomes more difficult.

A number of algorithms for star identification exist that can determine the correspondence between the viewed star field and a set of catalog stars in a known reference frame [4, 12, 13, 3, 5, 11, 10, 6, 7]. Despite this extensive

literature, and the apparent advantages of the use of a science camera for star identification, we are unaware of any algorithm that has demonstrated the capability of identifying stars from images produced by a sensor with a small FOV.

The grid algorithm for star identification was introduced in Padgett and Kreutz-Delgado [8]. This algorithm showed promise for accurate star identification using a medium sized FOV (8 degree diameter). Star identification algorithms employing star pair or triangle matching have great difficulty dealing with noise when identifying medium FOV images [9], while the grid algorithm performs quite well under these same conditions (99% correct identification with less than 1% false positives for expected levels of sensor noise). The on-board memory requirements and computation times for this algorithm are also acceptably small. The purpose of the new work is to enhance the grid algorithm to obtain satisfactory results on small FOV images (2 degree diameter).

In the current paper, we propose an extension to the grid algorithm that incorporates a Bayesian classifier to choose the star pattern from the on-board catalog that best matches the current sensor image and decide if the match is good enough to confidently identify it. Section 2 describes the algorithm including the extensions. Section 3 presents very promising results obtained by running the small FOV algorithm on a simulated sky containing approximately one million stars. Section 4 presents the discussion and conclusions.

2 Grid Algorithm

In this section we describe the grid star identification algorithm, so called because of the way the relative locations of stars within a field of view are represented. The representation used is called a *pattern.* First, we will describe how patterns are generated from a sensor image and the simple method used in [8] to test for similarity between two patterns. Next, we include a short discussion of how we go about simulating image noise, since an understanding of this is necessary for following the rest of the paper. After that, we will describe the new similarity measurement we have implemented based on Bayesian decision theory. Next, we will describe the information stored in the on-board star catalog and an efficient method for comparing a

sensor pattern against all patterns stored in the on-board catalog. Finally, we will show how several star identifications gathered from a single sensor image are combined to provide a decision about whether or not the position of the sensor image in the sky can be confidently ascertained.

2.1 Pattern Generation and Matching

A pattern is a representation of the relative locations of the stars within a FOV. From a single sensor image, it is possible to generate several distinct patterns, one for each star in the image. The steps for generating a pattern are listed below. Also see Figure 1.

- Choose a star from the sensor image to be the *center star.*
- Decide which star from the sensor image is the *neighbor star.* The neighbor star is the nearest star to the center star outside a radius of br pixels.
- Center a $g \times g$ grid on the center star, and orient the grid such that a horizontal vector from the grid center to the right edge passes through the neighbor star.
- Derive a g^2 element bit vector $pat\,[0 \ldots g^2 - 1]$ such that if grid $cell(i, j)$ contains a star, then $pat\,[j \times g + i] = 1$. All other elements of pat equal 0.

In this discussion, two parameters control the pattern generating process. The *grid resolution* parameter, g, specifies the number of cells per dimension of the grid that is laid over the sensor image. The *buffer radius*, br, determines how close a neighbor star can be to the center star. We set these parameters based on our experience with earlier simulations. For the simulations in this paper, $g = 80$ cells, and $br = 40$ pixels on a 1024×1024 pixel image plane.

A pattern may be generated not only from a sensor image, but from the star catalog as well. We can choose a star from the catalog to serve as the center star, project the positions of surrounding stars from the catalog onto a simulated sensor image, and generate a pattern as outlined above. The resulting pattern will be called a *catalog pattern.* A set of catalog patterns is stored in the on-board catalog, the database of star information kept in memory on the spacecraft.

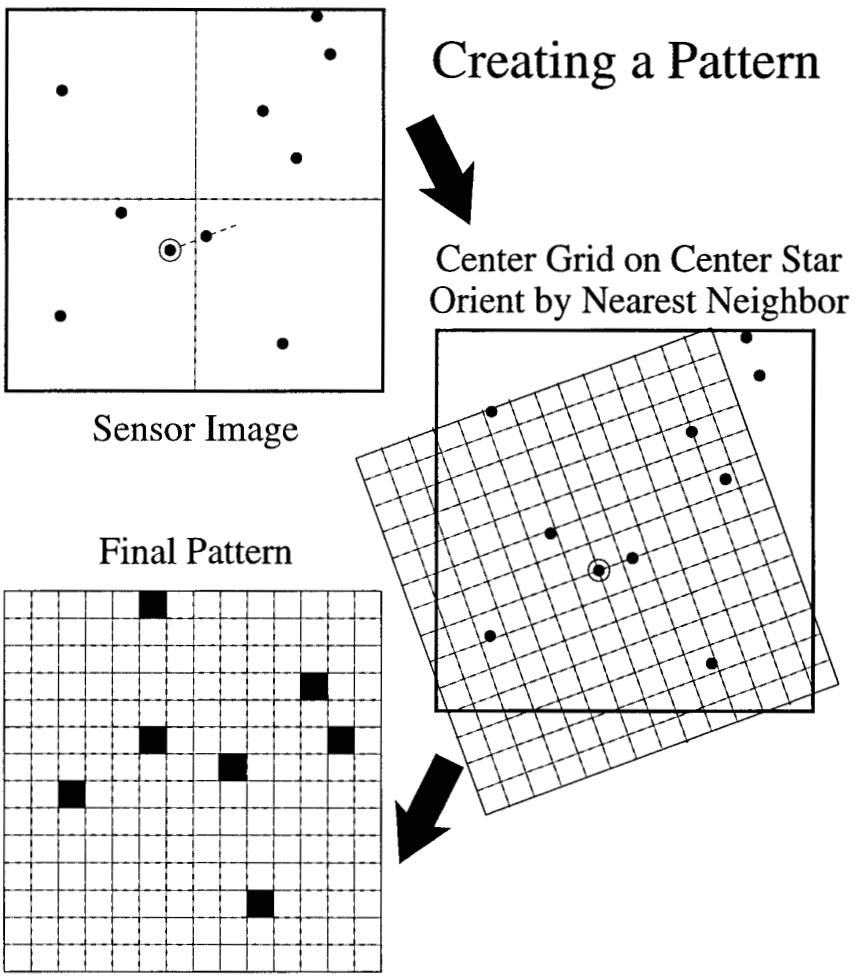


Figure 1: Creating a pattern from a sensor image. A grid is centered on the so-called *center star* and oriented so that the *neighbor star* is on the horizontal. Grid locations which contain a star are turned on.

A central process in the grid identification algorithm is finding the catalog pattern that is most similar to a specific sensor pattern. What does it mean for two patterns to be similar? If pat_i and pat_j are two patterns to be compared, we can count the number of non-zero cells the two patterns have in common with the following formula:

$$m(pat_i, pat_j) = \sum_{k=0}^{g^2-1} (pat_j[k] \times pat_i[k])$$

The catalog pattern which is most similar to pat_j, then, is the pat_i from the catalog which has the maximum value for $m(pat_i, pat_j)$.

The m metric gives a measure of how similar two patterns are. If the two patterns are similar enough, we may be confident that the sensor pattern center star is the same as the catalog pattern center star. In this case, it is said that the center star of the sensor pattern has been *identified.*

In an earlier version of the algorithm a constant threshold (T) was used to determine if the maximum value of m was good enough to identify a star. A value of m greater than or equal to T resulted in identification. For values less than T, we would say the star had an unknown identity. This worked quite well for medium FOV sensors [8] but proved inadequate for the small FOV sensors and resulting deep sky star catalogs we spoke of in the introduction. For this reason, we have developed the improved measure of similarity discussed in Section 2.3.

2.2 Image Noise

We cannot expect the star field images collected during a space flight to perfectly reproduce the view of the celestial sphere contained in our on-board catalog. Objects that do not appear in our catalog, such as planets and galaxies, may appear in the FOV. Portions of the sky can be obscured by the Earth or some other nearby body. Sensor optical properties may change the apparent position of objects which are in our catalog. Sensor noise may add spurious objects. Errors exist in the catalog regarding the brightness or position of listed stars. Variable or binary stars may change in magnitude and therefore not match the brightness values listed. Parallax effects may slightly change the positions of nearby stars relative to those farther away.

In our simulations, we model all of these possible errors using two types of Gaussian noise. During the production of a simulated image, zero mean

Gaussian noise may be added to the brightness of each star in the sensor image. Since it is measured in units of stellar magnitude, this is termed as *magnitude noise*. In our simulations, the simulated sensor is able to detect stars of 10.5 stellar magnitude or brighter. The addition of magnitude noise allows stars dimmer than magnitude 10.5 to appear in the sensor image and stars brighter than magnitude 10.5 to disappear. Zero mean Gaussian noise is also added to the position of each star in the sensor image. This type of noise, called *location noise* and measured in pixels, may cause stars to change grid locations in the sensor pattern or completely disappear off the edge of the sensor.

These two types of noise may result in significant changes in the appearance of a star image. For example, at 1.2 standard deviations of magnitude noise, and a 0.5 standard deviation of position noise, on average, 12.8 extra stars are added to the image, and 21.2 stars will disappear from the image[1]. The average number of stars seen in a sensor image at this high noise level in our simulations was 51.8.

2.3 Adaptive Threshold

In this section, we discuss the method used to determine if the number of stars which match between a sensor pattern and a catalog pattern (m) is large enough to consider the center star of the sensor pattern identified. In earlier versions of this algorithm, m was compared against a constant threshold (T) to decide whether or not to claim to have identified the sensor pattern center star. Using a constant threshold, however, ignores information which might be important in discriminating between correct and spurious matches, such as the density of stars on the sensor and the overlap between the catalog pattern and the sensor pattern.

The density of stars varies greatly across the celestial sphere (Figure 2). The probability of getting T spurious matches when there are 200 stars in the FOV is much higher than when there are 10 stars in the FOV. An algorithm that works poorly at either extreme faces a significant reduction in coverage of the celestial sphere. Using a threshold which adapts to the number of stars in the FOV provides a mechanism to increase the reliability of the grid

[1]More stars would have been added had the extracted catalog contained stars dimmer than magnitude 11.0.

algorithm at the extremes. We have implemented this adaptive threshold as a Bayesian classifier.

When we have chosen a star to be identified from the sensor FOV, we create a sensor pattern specific to that star by placing the star in the center of the pattern grid and rotating the grid so that the line from the center star to its nearest neighbor is horizontal (Figure 1). Since the pattern grid is no longer centered on the sensor, it is likely that a number of stars near the sensor edge will not appear in the sensor pattern. Likewise, there are areas on the pattern grid which are not covered by any area of the sensor. No stars will appear in these areas of the pattern. The *proportion of pattern overlap* is the proportion of pattern grid area which is covered by the sensor. The proportion of overlap for catalog patterns is always 1.0.

In the following discussion, the proportion of pattern overlap is designated by the variable, o. The variable, s, we will use to designate the number of stars in the current sensor pattern. Similarly, t will represent the number of stars in the current catalog pattern. As we mentioned before, the number of matching grid cells found between the sensor pattern and the current catalog pattern will be given by m.

If the center star of the current catalog pattern is the same star as the center star of the sensor pattern, and the nearest neighbor star in both patterns is the same, we will say that we have a *correct* identification. If the center star of the catalog pattern is wrong or the nearest neighbor is wrong, we will call that identification *incorrect*. Let $P(correct \mid m, s, t, o)$ be the probability of seeing a correct identification given the current parameter settings, m, s, t, and o. Similarly, $P(incorrect \mid m, s, t, o)$ is the probability of seeing an incorrect identification given the current parameters.

If we set the adaptive threshold at the point where

$$P(correct \mid m, s, t, o) = P(incorrect \mid m, s, t, o). \tag{1}$$

we will minimize the average probability of misclassification [2][2]. If $P(correct \mid m, s, t, o) > P(incorrect \mid m, s, t, o)$, we will tag the sensor pattern center

[2]This analysis assumes that the cost of a misclassification is the same regardless of whether the error is a false positive or a false negative. In this setting, this assumption is very likely wrong, since identifying an orientation incorrectly may result in a failed mission, while reporting an unknown orientation simply requires acquiring an image and trying again. Setting the threshold to a point where, for example, $P(correct \mid m, s, t, o) > P(incorrect \mid m, s, t, o)$ may improve the performance of the algorithm by reducing the number of false positives. On the other hand, the verification phase of the algorithm

star with the identification number of the matching star from the on-board star catalog. Otherwise we will tag the center star as unknown. By Bayes theorem, we can rewrite (1) as

$$P(m, s, t, o \mid correct)P(correct) = P(m, s, t, o \mid incorrect)P(incorrect) \quad (2)$$

From the definition of conditional probability, $P(x \mid y) = \frac{P(x,y)}{P(y)}$, we get $P(x, y) = P(x \mid y)P(y)$. We can use this identity to rewrite (2) as

$$P(m \mid s, t, o, correct)P(s, t \mid o, correct)P(o \mid correct)P(correct) =$$
$$P(m \mid s, t, o, incorrect)P(s, t \mid o, incorrect)P(o \mid incorrect)P(incorrect). \quad (3)$$

The proportion of overlap is independent of whether we have a correct identification or not, so $P(o \mid correct) = P(o \mid incorrect)$. Therefore, (3) can be rewritten as

$$P(m \mid s, t, o, correct)P(s, t \mid o, correct)P(correct) =$$
$$P(m \mid s, t, o, incorrect)P(s, t \mid o, incorrect)P(incorrect). \quad (4)$$

Then we apply the identity again to produce

$$P(m \mid s, t, o, correct)P(s \mid t, o, correct)P(t)P(correct) =$$
$$P(m \mid s, t, o, incorrect)P(s, t \mid o, incorrect)P(incorrect). \quad (5)$$

Let us look at the right side of this equation first. We have an incorrect identification here, so the sensor and catalog patterns are generally not from the same part of the sky. We will make the assumption, then, that the numbers of stars in the sensor and catalog patterns are independent [3]. This allows us to rewrite $P(s, t \mid o, incorrect)$ as $P(s \mid o, incorrect)P(t \mid o, incorrect)$. Let us also assume that fact that we have an incorrect identification does not

(described in Section 2.5) tends to filter out the majority of false positives anyway, so this change in the threshold may end up hurting performance by reducing the number of correct identifications. We have not yet attempted an analysis which includes the influence of the verification phase or the costs of the various types of errors, so we simply assume equal cost.

[3]Note that, when the catalog pattern is correct but the nearest neighbor chosen is incorrect, the identification is considered incorrect. In this case, the two patterns are from the same part of the sky and so the number of stars is highly correlated. The assumption we are making here ignores this situation.

influence the number of stars on the sensor[4], so $P(s \mid o, incorrect) = P(s \mid o)$. If we assume that the stars are uniformly distributed on the sensor, we get $P(s \mid o, incorrect) = oP(s)$, where $P(s)$ is the probability of getting s stars in a sensor pattern when the proportion of overlap is 1. This is essentially equivalent to the distribution of star densities across the sky.

Similarly, we can rewrite $P(t \mid o, incorrect)$ as [5] $P(t \mid o)$, since the number of stars in the catalog pattern is independent of the proportion of overlap, $P(t \mid o, incorrect) = P(t)$. Putting this all together, we see that $P(s \mid o, incorrect)P(t \mid o, incorrect) = oP(s)P(t)$.

Let us consider another term on the right side of (5), $P(m \mid s, t, o, incorrect)$. For the sake of simplicity, when we have an incorrect identification, we will assume that any match we get between the sensor pattern and a catalog pattern is purely random, and that each potential match is independent of all others. For each star in the sensor pattern, the probability that it matches a star in the current catalog pattern by chance is $\frac{t}{g^2}$, where g is the number of cells per dimension in the grid. This gives us a binomial distribution $B(s, \frac{t}{g^2}, m)$, where $B(x, y, z)$ equals the probability of getting z successes out of x independent trials when the probability of success on each trial is y. Thus, we can rewrite (5) as

$$P(m \mid s, t, o, correct)P(s \mid t, o, correct)P(t)P(correct) = B(s, \frac{t}{g^2}, m)oP(s)P(t)P(incorrect). \quad (6)$$

Simplifying, and using the fact that $P(incorrect) = 1 - P(correct)$, gives us

$$P(m \mid s, t, o, correct)P(s \mid t, o, correct)P(correct) = B(s, \frac{t}{g^2}, m)oP(s)(1 - P(correct)). \quad (7)$$

Equation 7 gives us four probability distributions which must be estimated, $P(s)$, $P(correct)$, $P(s \mid t, o, correct)$, and $P(m \mid s, t, o, correct)$. We

[4] Of all our assumptions, this one seems most suspect. Despite the dubious quality of this assumption, the algorithm does perform well. A more complex analysis which does not make this assumption is possible. We have not done any empirical studies to determine what influence, if any, such a change would have on performance.

[5] Strictly speaking, the fact that a catalog pattern is incorrect does influence the distribution of the number of stars in the pattern, because there is one pattern we are less likely to see when we have an incorrect identification – the correct catalog pattern. Given the large number of catalog patterns, this influence is negligible.

generated an estimation for each of these distributions by running simulations. Recall that $P(s)$ is essentially equivalent to the distribution of star densities across the sky. We estimated this distribution by counting the number of stars within a 2 degree FOV for 10,000 random sky orientations and plotting the count of times each number of stars occurred. A smoothed version of this plot, normalized so that the area under the curve is 1.0 (Figure 2), serves as a good estimate of $P(s)$.

We estimated $P(correct)$ by running our algorithm on 1000 randomly chosen sky orientations and counting the number of times an identification occurred. The proportion of identifications estimated by this method was 0.22.

When we have a correct identification, the sensor and catalog patterns are taken from the same part of the sky, and we should expect a high degree of correlation between the counts of stars in the two patterns. This correlation must be captured by our estimate of the distribution $P(s \mid t, o, correct)$. When we have a correct identification, a good first estimate of s is given by $s = to$. For each integral value of $\lfloor to \rfloor$ we calculated the mean value of s collected from 500 randomly generated correct identifications at magnitude noise of 1.0 and location noise of 0.5. A plot of to versus the mean of s can be fitted by a line passing through the origin with slope 0.64.[6] Given this understanding of the distribution, we decided to estimate $P(s \mid t, o, correct)$ by the probability of seeing the value $s - 0.64to$ when choosing from the normal distribution with mean 0. The variance of the normal distribution (134) was estimated by calculating the variance of to across the previously mentioned 500 randomly generated orientations[7].

[6] If $s = to$, we would see a slope of 1. The bias towards smaller values of s is caused by the fact that our simulated sky contains no stars of magnitude greater than 11. Our simulated sensor can detect stars up to magnitude 10.5. Adding Gaussian noise to the magnitude of the stars will cause a number of stars brighter than 10.5 to become dimmer than the sensor can detect, and thus disappear. Also a number of stars dimmer than 10.5 will become brighter, and thus appear. With Gaussian noise of 1.0 standard deviations, as we are using in these simulations, we would expect a number of stars dimmer than magnitude 11 to appear on the sensor. Since there are no such stars in our simulated sky, this does not happen. Some of the stars which vanish because of the noise are not replaced, and the number of stars on the sensor, s, decreases.

[7] A plot of to versus variance shows that the variance is clearly not constant. It is vastly smaller for small values of to. Changing our estimate of $P(s \mid t, o, id)$ to take this into account may improve performance of the algorithm in sparse star fields.

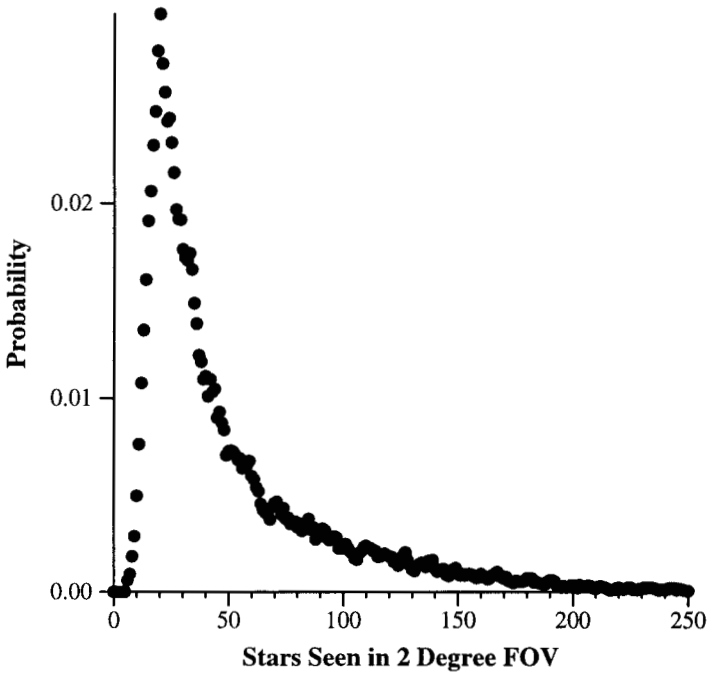


Figure 2: Probability of seeing various numbers of stars in a 2 degree FOV sensor image.

There remains one distribution to estimate, $P(m \mid s, t, o, correct)$. This distribution tells how likely we are to get m matches given the number of stars in the sensor pattern, the current catalog pattern, and the proportion of overlap between the two patterns. Since we are dealing here with a correct identification, if there were no noise, we would expect every star in the sensor pattern to match the corresponding star in the catalog pattern (i.e., $m = s$).

We used a binomial distribution to estimate $P(m \mid s, t, o, incorrect)$, and we employ a similar strategy here to estimate $P(m \mid s, t, o, correct)$. We assume that there are r stars in the area of the sky from which we are generating a sensor image and that for each of these stars there is a probability, p, that it will appear on the sensor. Note that we do not know either r or p, so these values must be estimated. Since we have a correct identification, s and to both serve as estimates of r. We use the average, $\frac{s+to}{2}$.

The value for p is estimated by simulation. For each of 1000 randomly chosen catalog stars, we center the sensor on the star and create a sensor

pattern after adding 1.0 standard deviation of magnitude noise and 0.5 of location noise to all the stars. Under this procedure, only 52% of the stars end up in the same grid location as they occupy in the corresponding catalog pattern, so our estimate of p is 0.52.

The noise we use in our simulations can affect the value of m in three ways. First, magnitude noise may increase or decrease the brightness of a star as it appears on the sensor. This may cause some stars to appear and others to disappear. Second, location noise may cause a sensor star to change grid locations so that a match with its corresponding catalog star may be missed. Both of these effects are consistent with using the binomial distribution as an estimate of $P(m \mid s, t, o, correct)$. The third effect is not. Location noise on the center star or its nearest neighbor may cause the alignment of the sensor pattern to rotate compared to the catalog pattern. Note that the farther a star is from the center star the more likely it is to change grid locations, reducing the number of matches, m. The probabilities of two stars appearing in the correct grid location are not independent under this effect. If location noise causes a large rotation, all stars in the pattern are affected.

Clearly, this third effect violates the assumptions of using the binomial distribution as an estimate of $P(m \mid s, t, o, correct)$. To more accurately reproduce the actual distribution, we could condition this probability on the distances from the center star to each other star in the sensor image. This would complicate things considerably, so we decided to use the simpler binomial model. The results reported in Section 3 support our decision.

We started with Equation 1, which contained two distributions whose shapes were unknown. Putting everything together, we end up with Equation 8, which contains only distributions which are completely defined.

$$B(\frac{s+to}{2}, 0.52, m)N(0, 134, s = 0.64to)0.22 = B(s, \frac{t}{g}, m)oP(s)0.78 \quad (8)$$

where $N(x, y, z)$ is the height at z of the normal distribution with mean x and variance y.

Equation 8 defines a threshold which changes depending on the values of s, t, and o. If the right side of Equation 8 is greater than the left, then we will label the center star of the sensor pattern with the label of the center star of the catalog pattern. If the left side is greater, we will label the center star as "unknown." If we subtract the left side from the right, as in Equation 9, we get an adaptive measure of how well the two patterns match. The larger

the number, the better the match.

$$B(\frac{s+to}{2}, 0.52, m)N(0, 134, s - 0.64to)0.22 - B(s, \frac{t}{g}, m)oP(s)0.78 \quad (9)$$

2.4 On-Board Star Catalog

If the algorithm is to be effective for all positions in the sky, we need to insure that the on-board catalog contains patterns for some minimal number of center stars in each potential sensor image. Furthermore, in order to increase the likelihood that the algorithm will be able to identify all potential orientations with the same facility, it is necessary for each potential orientation to have approximately the same number of center stars in the catalog.

To achieve these goals, we use a relatively simple procedure for determining which center stars to include in the on-board catalog. Typically the brightest stars are more reliably imaged and extracted than are the dimmer stars, so using the bright stars should result in increased accuracy. We choose a set of orientations covering the celestial sphere by walking the sphere in 0.25 degree incremental steps. From each orientation generated, the brightest stars within a radius of pr degrees are added to the catalog until a total of α stars within that radius are included, or until we run out of stars.

The parameter pr, or *pattern radius* is chosen to approximately cover the radius of the sensor image. In the experiments presented here, we will be simulating a square sensor image of 2 degrees per side. A circle of 1.4 degree radius approximately circumscribes such an image, and so we set pr to 1.4. The parameter, α approximates how many recognizable center stars we would like to see in a sensor image. Here, we continue the tradition established in earlier research of setting α to 10. Using these parameter values, this method gives us a fairly uniform scattering of approximately 100,000 bright center stars across the celestial sphere.

For each center star chosen by this method, we create an entry in the on-board catalog. The index of this entry serves as the identification number of the center star. Each entry contains the position (right ascension and declination) of the center star on the celestial sphere, and the number of ones in the pattern generated from the center star. This last value is used as the t value for the calculations which were presented in Section 2.3.

The pattern for a center star is not stored in the bit vector form described in Section 2.1. Rather than storing each pattern separately, an aggregate

structure, called a lookup table, is used, which allows a fast search for the best match across all patterns of the on-board catalog. For each of the g^2 pattern grid locations, we store a list of the identification numbers for the center stars whose patterns have a star at that location.

To find the center star whose pattern best matches a sensor pattern, pat_j, we simply examine the lookup table at each bit location in pat_j where a one occurs, and increment a counter for each center star listed there. At the end of this procedure, each counter will contain the number of matching stars, m, found for the corresponding catalog pattern. The catalog pattern counter with the highest value tells us the on-board pattern which best matches pat_j (Figure 3).

2.5 Processing a Star Field Image

Once the on-board catalog has been constructed, the actual identification process is quite simple. The input to the grid algorithm is a sensor image. A star in the sensor image has information regarding its position on the sensor and its apparent brightness. Ideally, the brightest $\alpha = 10$ stars in the sensor image would correspond to 10 stars which had been included in the on-board catalog for this orientation. Due to noise, however, some of the on-board catalog stars may not be among the brightest in the image. For this reason, we test up to 3α of the sensor stars. If we were to test all of the sensor stars, we would increase the risk of generating spurious matches (maximum matches of sensor stars with the wrong on-board catalog star).

For the grid algorithm to work, it is essential that the same nearest neighbor star is found in the image as was used in generating the on-board pattern. To increase the probability of finding the correct nearest neighbor, if we fail to find a good enough match using the nearest neighbor, we restart the matching process a second time using the second nearest neighbor. This procedure is used for both the static and adaptive threshold algorithms.

Ideally, with the new adaptive threshold algorithm, we would evaluate each catalog pattern using Equation 9 (from Section 2.3). The catalog pattern which produced the largest value would be the best match. Since this would be a time consuming process, instead we find the β catalog patterns with the largest numbers of matches, m, and evaluate only these β patterns using Equation 9. If the maximum value of Equation 9 is greater than zero, we label the sensor pattern center star with the identifier of this best match-

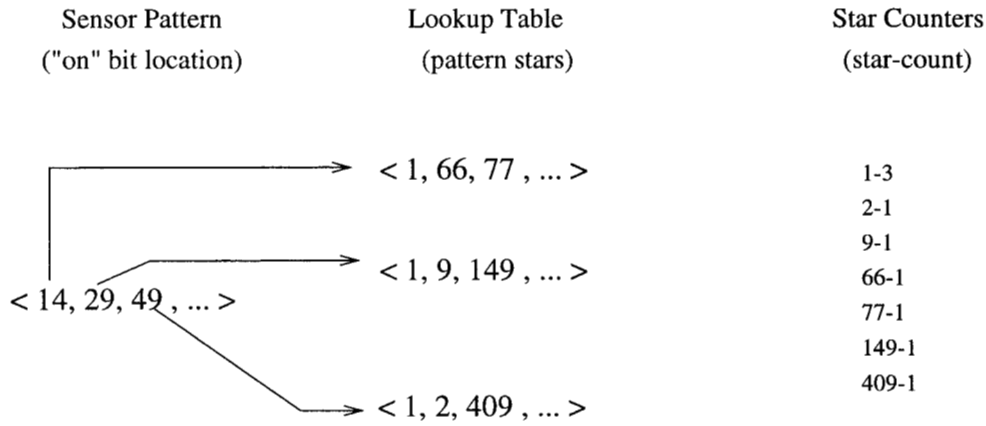


Figure 3: The leftmost column of this figure shows part of the sensor pattern representation extracted from an image, indicating that stars were detected in grid cell locations 14, 29, 49, etc. In the center column, we see how similar information was recorded for catalog patterns in the on-board lookup table. Lookup table entries for grid locations 14, 29, and 49 all contain a 1, indicating that catalog pattern number 1 contains stars in these grid locations. Note that other patterns also contain stars in these locations. For instance, catalog patterns 66 and 77 contain a star in grid location 14. The rightmost column shows the set of star counters which have so far been incremented by this sensor pattern. Note that the star counter for catalog pattern 1 has been incremented 3 times, once for each time pattern 1 showed up in one of the lookup entries corresponding to a sensor pattern "on" location. Other star counters (2, 9, 66, etc.) have also been incremented, but the largest number of matches found so far (3) are for pattern number 1.

ing catalog pattern.

We choose a value of β which is likely to include the best matching catalog pattern, according to Equation 9, but which will not require too much computation. Note that changing this value will affect the value of $P(correct)$ discussed in Section 2.3. In the simulations reported here, β was set to 7.

Regardless of whether we use the static or adaptive threshold algorithm, at this point we have labeled up to $3\alpha = 30$ sensor stars with on-board catalog indices. We can now assign positions to each of these stars from the on-board catalog. It is quite likely that some stars will be mislabeled. Therefore, we perform a final verification operation to test the consistency

of the labels. Any labeled star which is not near any other labeled star is obviously from the wrong part of the sky. The largest group of stars within the same FOV is located. If the number of stars in this group is greater than or equal to two, then the orientation of the boresight of the image is calculated from the positions of the labeled stars. Otherwise, the orientation of the image is reported as unknown.

3 Simulation Results

To determine how well the grid algorithm performs, a number of simulations were conducted to measure the identification rate under a variety of different noise conditions. In this section, we present the results of these simulations for both the static threshold and adaptive threshold algorithms.

For these experiments, we simulated the celestial sphere by creating a star catalog which includes all stars brighter than magnitude 11.0 extracted from the Guide Star Catalog Version 1.1. The extracted catalog contains 934,487 stars.

All star images were created using a simulated sensor with a 2 degree $\times$ 2 degree square field of view, projecting onto a square CCD array of 1024 $\times$ 1024 pixels. The simulated sensor was capable of detecting stars brighter than magnitude 10.5.

The on-board catalog was created from the extracted catalog as described in Section 2.4. No noise was added to the catalog during this process.

3.1 Static Threshold Algorithm

To see the effect of changes in the static threshold T on the performance of the static threshold algorithm, we ran a series of simulations. The results are reported in Figure 4. Each data point reports the proportion of correct, incorrect, or unknown orientation identifications seen across 500 randomly generated orientations for one setting of T. A correct orientation occurs when the algorithm returns the estimated position of an image and it is correct. An incorrect orientation is when the algorithm returns the position and it is incorrect. The remainder of the cases are reported as unknown orientations. These occur when the algorithm reports that the position is unknown. Location error in these simulations is kept at a constant value

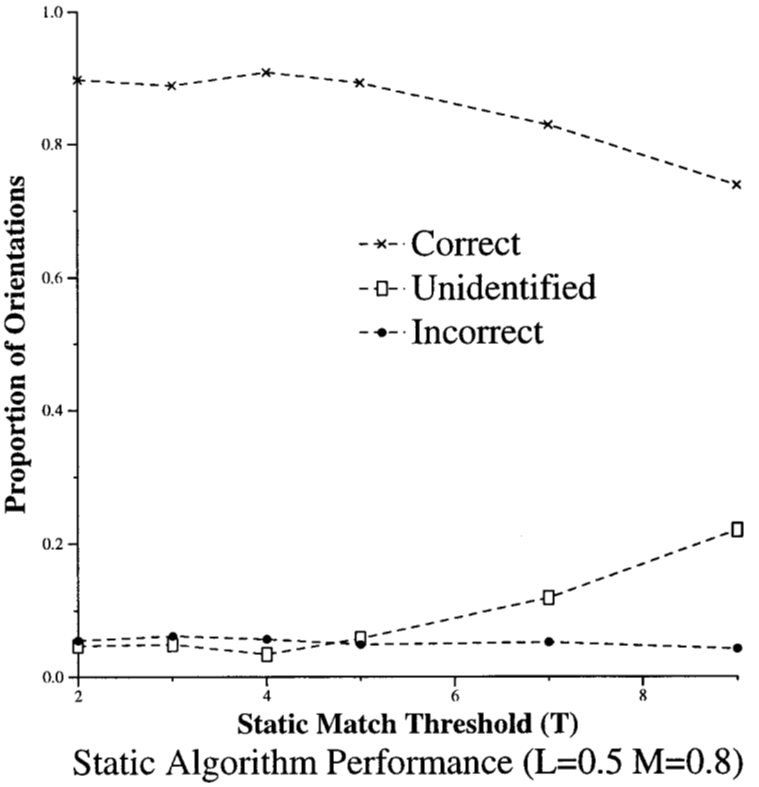


Figure 4:

of 0.5 pixels standard deviation. Magnitude error is set at 0.8 units stellar magnitude standard deviation.

As the threshold, T, is raised, we expect more unlabeled stars to be sent into the verification phase. This should result in more unidentified orientations and fewer correct and incorrect orientations. This is the trend displayed in the graph. It may seem surprising that a change in the threshold has so little effect on the number of incorrect orientations. The flat slope of this curve demonstrates that the verification phase is quite effective at finding and ignoring incorrectly identified stars.

The highest percentage of correct orientations (90.9%) is seen when T is set to 4. At this threshold value, the percentage of unknown and incorrect orientations is 3.4% and 5.7%, respectively. This best threshold value is quite low compared to the values used in previous, larger FOV simulations.

This lowering of the threshold is due to the increased effectiveness of the verification phase as the FOV gets smaller, as predicted in [8]. As the FOV gets smaller, the probability is reduced that two random orientations will, by chance, fall into the same FOV. This allows the verification phase to ignore more spuriously labeled stars.

The proportion of incorrect orientations is a very important measure of how well the algorithm is performing. If a spacecraft is using the algorithm to restore its (lost) orientation knowledge from a single star field image, then an incorrect identification can be disastrous. If the spacecraft performs maneuvers using the incorrect orientation knowledge, it is likely to become even more lost. On the other hand, if the algorithm reports an unknown orientation, the spacecraft can reorient and try again with another star field image.

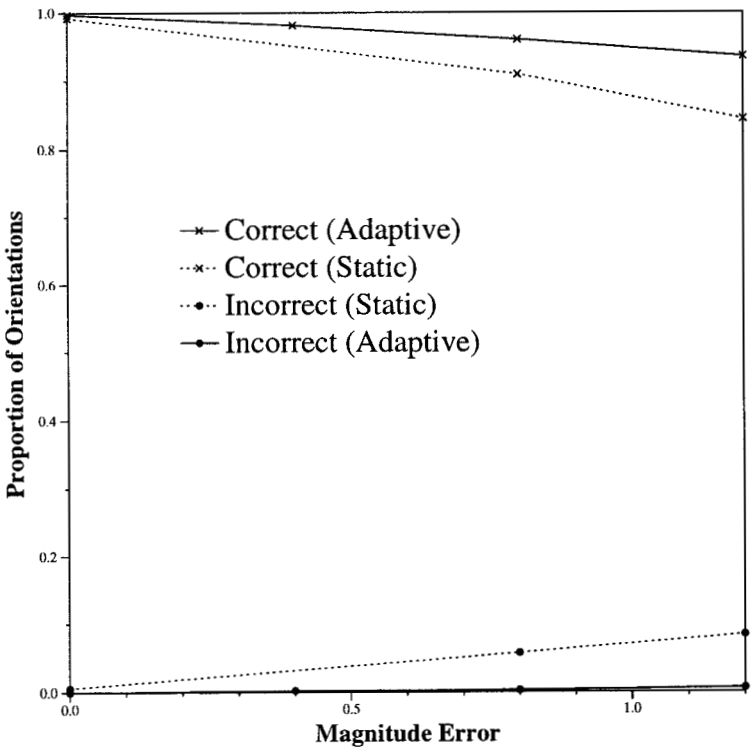
The results reported here are quite good considering the level of noise and the small field of view we are dealing with. Nevertheless, as we will see in the next section, even better results are possible with the adaptive threshold algorithm.

3.2 Adaptive Threshold Algorithm

In this section, we present the results of a set of simulations which demonstrate the accuracy of the grid algorithm using the adaptive threshold technique explained in Section 2.3.

Figure 5 contrasts the accuracy of the adaptive threshold algorithm (solid line) under varying levels of magnitude noise with the accuracy of the static threshold algorithm (dotted line) with $T = 4$ under the same conditions. The x axis tells the level of magnitude noise, which ranges from 0.0 to 1.2 standard deviations. A constant 0.5 pixel of position noise was injected across all simulations. Each data point reports the proportion of correct or incorrect orientation identifications seen across 500 randomly generated orientations for the static algorithm, and at least 1000 randomly generated orientations for the adaptive algorithm. To simplify the graphs, the third measure, unknown orientations, is not reported here. The sum of all three possibilities equals 1.0, so the proportion of unknown orientations could be extracted from the reported values.

Notice that at all levels of noise, the adaptive threshold algorithm produces a higher proportion of correct identifications, while maintaining a much



Adaptive Algorithm Performance vs. Static (L=0.5)

Figure 5:

lower number of incorrect identifications. For example, at the 0.8 level of magnitude noise, 96.0% of orientations are correctly identified with an incorrect rate of less than 0.3%. The static algorithm produced 90.9% correct and 5.7% incorrect at this noise level.

The effect of location noise on the adaptive algorithm is demonstrated in Figure 6. The x axis tells the level of location noise, which varies from 0.0 to 1.5 standard deviations measured in pixels. A constant level of magnitude noise (0.4 standard deviation) was injected across all data points. Each data point is collected across 500 randomly generated orientations. The small slope on all the curves suggests that reasonable levels of location noise have a smaller effect on performance than does magnitude noise. This is consistent with our previous observations with the static algorithm. At 1.0 pixel of location noise, a rather high level, the proportion of correct identifications

is 97.0%, while incorrect identifications are still quite low at 0.4%, leaving 2.6% unidentified.

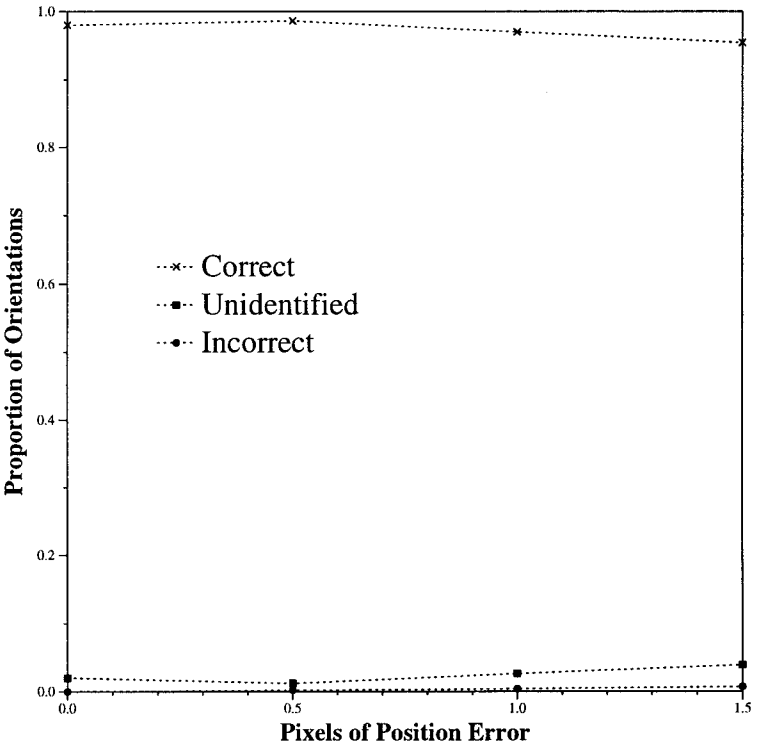
To our knowledge, there exist no published results on the performance of any star identification algorithm using such a narrow FOV. Padgett, Kreutz-Delgado, and Undomkesmalee (1997) compare the performance of three competing algorithms on a much larger field of view — a triangle algorithm, a match group algorithm, and the grid algorithm with static thresholding[8]. Comparing the results of the grid algorithm with adaptive thresholding, as reported above with those of Padgett et al. indicates that this new algorithm is much more robust in the face of noise than either the triangle or match group algorithms even though the test of the competing algorithms was based on an 8 degree FOV sensor, while the grid algorithm with adaptive thresholding was tested at 2 degrees.

3.3 Memory and Computational Requirements of the Algorithm

This section presents the memory and computational requirements of the grid algorithm with adaptive thresholding.

The only static data structures maintained by the algorithm are the on-board catalog, and the look-up table, discussed in Section 2.4. These two structures account for the large majority of the memory required to run the algorithm. The amount of memory consumed by these structures is quite easy to calculate. Each on-board catalog entry contains the right ascension and declination of the center star, and the number of stars in the pattern generated from the center star. Assuming each of these values requires four bytes of storage, the amount of memory required for the on-board catalog is $12n$, where n is the number of entries in the on-board catalog. Each lookup table entry consists of a list of on-board catalog indices for the center stars which contain a 1 in a particular grid location. Each of these lists can compactly be stored in an array. If the average number of stars per pattern associated with an on-board catalog entry is given by a, then the total number

[8]The comparison concentrates on the identification phase of the various algorithms, so all algorithms use simple verification phases similar to that used in the simulations reported here. We expect use of a more complicated verification phase to boost the performance of all three algorithms.



Adaptive Algorithm Performance at 0.4 Magnitude Error

Figure 6:

of indices stored in the lookup table is na. Assuming each index can be stored in 4 bytes, the total memory requirement of the lookup table is $4na$, and the memory required for both static structures is $n(12+4a)$. In our simulations, $n = 101,490$ and $a = 56.0$, making the total memory requirements for the static data structures in this instantiation of the algorithm approximately 24 megabytes.

The number of seconds of CPU time per orientation calculated across 5000 orientations has been recorded at approximately 8.5 seconds per orientation on a Sun SPARCstation 10. This figure does include a certain amount of overhead, mainly gathering of statistics, which would not be present in a flight-ready implementation.

4 Conclusions

Despite the good results we have demonstrated here, we expect that performance improvements are still possible by future enhancements of the algorithm. Some improvement in performance is likely necessary in order to push the FOV down to the diameter of many science cameras (1 degree or less).

Using a sensor capable of detecting even dimmer stars should improve performance somewhat. This is likely to be quite important at smaller fields of view. At the sensitivity level at which the current simulations were run (10.5 magnitude), there are still quite a few orientations with very few stars, as few as one. We expect that if it is possible to accurately image even dimmer stars, this will increase the number of correct identifications in these sparse areas.

As mentioned in Section 2.3, noise which causes the location of the center star or its nearest neighbor to change slightly in the sensor image may introduce a twist in the alignment of the grid, effectively changing the location of all the stars in the image relative to the catalog pattern. The square grid used in these experiments is clearly not optimal for dealing with this problem since a small twist is likely to change the grid location of all stars which are far away from the center. A target-shaped grid composed of concentric circles divided into sectors by lines radiating from the center should improve the ability of the algorithm to deal with this kind of noise. If such a grid is centered on the center star, a twist about the center will have a similar effect on all stars regardless of their location on the grid.

A number of competing star identification algorithms include sophisticated verification phases to remove spurious star identifications [4, 13]. Incorporation of such a technique into our algorithm may result in performance levels unachievable with the simple verification technique used here.

As mentioned in Footnote 2, a deeper analysis, which incorporates the influence of the verification phase, or which takes into account the cost of the various types of errors, may result in a better setting of the adaptive threshold, resulting in improved performance.

In addition to improving the performance of the algorithm, we would like to try out the algorithm on real star images. The noise models used in this paper are fairly typical of this kind of research. Despite this, they, no doubt, fall short of capturing all the important characteristics of actual image noise. Unfortunately, star field images at very small fields of view are not readily

available. Access to an observatory may be required to make progress in this direction.

The simulations presented in this paper provide convincing evidence that the adaptive threshold grid algorithm is capable of acceptable performance for typical star identification applications, even using a small FOV science sensor with fairly high levels of sensor noise. To our knowledge, this work represents the first successful attempt to tackle this difficult problem.

References

[1] Association of Universities for Research in Astronomy, Inc. The guide star catalog, version 1.1, 1992. CD ROM available from Space Telescope Science Institute, Baltimore, MD.

[2] Richard O. Duda and Peter E. Hart. *Pattern Classification and Scene Analysis.* John Wiley & Sons, New York, 1973.

[3] E. Groth. A pattern-matching algorithm for two-dimensional coordinate lists. *Astronomical Journal*, 91:1244–1248, 1986.

[4] J. Junkins, C. White, and D. Turner. Star pattern recognition for real time attitude determination. *Journal of the Astronautical Sciences*, 25:251–270, 1977.

[5] J. Kosik. Star pattern identification aboard an inertially stabilized spacecraft. *Journal of Guidance, Control and Dynamics*, 14(2):230–235, 1991.

[6] C. Liebe. Pattern recognition of star constellations for spacecraft applications. *IEEE Aeronautics and Electronic Systems Magazine*, June 1992.

[7] F. Murtagh. A new approach to point-pattern matching. *Astronomical Society of the Pacific*, 104:301–307, April 1992.

[8] C. Padgett and K. Kreutz-Delgado. A grid algorithm for star identification. *IEEE Aerospace and Electronics Systems*, Jan 1997.

[9] C. Padgett, K. Kreutz-Delgado, and S. Undomkesmalee. Evaluation of star identification techniques. *Journal of Guidance, Control and Dynamics*, 20(2), 1997.

[10] B. Sheela and C. Shekhar. New star identification technique for attitude control. *Journal of Guidance, Control and Dynamics*, 14:477–480, 1991.

[11] T. Strikwerda, H. Fisher, C. Kilgus, and L. Frank. Autonomous star identification and spacecraft attitude determination with CCD star trackers. In *1st European Space Agency International Conference on Spacecraft Guidance, Navigation and Control Systems*, Noordwijk Netherlands, 1991.

[12] T. Strikwerda and J. Junkins. Star pattern recognition and spacecraft attitude determination. Technical Report ETL-0260, Engineer Topographic Laboratories, 1981.

[13] R. van Bezooijen. *Automated Star Pattern Recognition.* PhD thesis, Stanford University, 1989.

[14] J. Wertz. *Spacecraft Attitude Determination and Control.* D. Reidel Publishing Co., MA, 1978.